Journal of Biomedical Informatics 46 (2013) S13–S19

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Extraction of events and temporal expressions from clinical narratives

Prateek Jindal *, Dan Roth

Department of Computer Science, UIUC, 201 N. Goodwin Ave, Urbana, IL 61801, United States

ARTICLE INFO

Article history:
Received 12 March 2013
Accepted 27 August 2013
Available online 8 September 2013

Keywords:
Natural language processing
Named entity recognition
Electronic health records
Information extraction
Temporal extraction
Integer quadratic programmming

ABSTRACT

This paper addresses an important task of event and timex extraction from clinical narratives in context of the i2b2 2012 challenge. State-of-the-art approaches for event extraction use a multi-class classifier for finding the event types. However, such approaches consider each event in isolation. In this paper, we present a sentence-level inference strategy which enforces consistency constraints on attributes of those events which appear close to one another. Our approach is general and can be used for other tasks as well. We also design novel features like clinical descriptors (from medical ontologies) which encode a lot of useful information about the concepts. For timex extraction, we adapt a state-of-the-art system, HeidelTime, for use in clinical narratives and also develop several rules which complement HeidelTime. We also give a robust algorithm for date extraction. For the event extraction task, we achieved an overall F1 score of 0.71 for determining span of the events along with their attributes. For the timex extraction task, we achieved an F1 score of 0.79 for determining span of the temporal expressions. We present detailed error analysis of our system and also point out some factors which can help to improve its accuracy.

1. Introduction

The i2b2 2012 challenge [1] was focussed on temporal information extraction from clinical narratives. This challenge consisted of three tasks: (a) event extraction, (b) temporal expression (or timex) extraction and (c) extraction of temporal relations. We participated in the first two tasks, namely, event and timex extraction. This paper describes our approach to extracting events and temporal expressions from clinical narratives. Input for these tasks consists of clinical narratives (in plain text format) and goal is to automatically annotate the clinical narratives with events and timexes information. Along with finding text spans of events and timexes, these tasks also involve finding their attributes.

A sub-task of event extraction (i.e. concept extraction) for clinical narratives has been previously addressed in some of the works [2–7]. Uzuner et al. [8] give a survey of systems that participated in the 2010 i2b2 NLP challenge. Best systems for this task used machine learning approaches to find concept boundaries and types. Most of the systems used some sequence prediction model in conjunction with a multi-class classifier (like Support Vector Machines (SVMs)) for finding concept boundaries and types. Sequence prediction models that are typically used are first-order models so that the inference remains tractable. Such models cannot account for expressive (long-range) dependencies between different events. In this paper, we present a general inference strategy which can easily account for such expressive dependencies.

For timex extraction, best available systems (like HeidelTime [9] and SUTime [10]) are rule-based. HeidelTime is a multilingual cross-domain temporal tagger and was the best system for extraction and normalization of English temporal expressions in the TempEval-2 [11] challenge. SUTIME is another temporal tagger and is available as part of the Stanford CoreNLP pipeline. Testing on the TempEval-2 evaluation corpus shows that this system gives results comparable to HeidelTime. Angeli et al. [12] present a probabilistic approach (using a Probabilistic Context Free Grammar (PCFG)) for learning to interpret temporal phrases. However, this system does not perform as well as HeidelTime and SUTime on TempEval-2 task. Both HeidelTime and SUTime have been designed for general English text and are not sufficient for extracting clinical timexes. We developed rules specific to clinical narratives which complement HeidelTime for the task of clinical timex extraction. Sun et al. [13] provide comprehensive overview of systems which perform clinical timex extraction.

The primary contributions of this paper are the following:

1. *Novel Features*: We designed novel features for event extraction. One of our features is to construct a clinical descriptor for any concept using medical ontologies (like MeSH and SNOMED CT). The clinical descriptors designed by us encode important information about concepts and would benefit several other Information Extraction tasks. For example, clinical descriptor of the concept "Myocardial Infarction" (or Heart Attack)

* Corresponding author.
E-mail addresses: jindal2@illinois.edu (P. Jindal), danr@illinois.edu (D. Roth).
URLs: http://www.cogcomp.cs.illinois.edu/~jindal2/ (P. Jindal), http://www.l2r.cs.uiuc.edu/ (D. Roth).

1532-0464/$ - see front matter
http://dx.doi.org/10.1016/j.jbi.2013.08.010

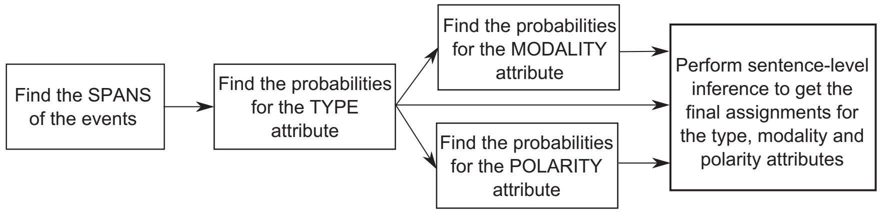


Fig. 1. Pipeline approach for finding event boundaries and attributes.

consists of the following terms: Disease, Traumatic Injury, Disease affecting entire cardiovascular system, Myocardial Ischemia, Heart Disease, etc.

2. *Sentence-Level Inference*: Attributes of events which appear close to one another are sometimes related. To use such information, we developed an inference strategy which ensures that the attributes of related events are consistent with one another. Our approach advances the current state-of-the-art for event extraction where events are considered in isolation while determining their attributes. Moreover, our approach is general and can also be used for other cases where one is jointly solving several Information Extraction tasks which constrain one another.
3. *Date-extraction algorithm and rules for timex extraction*: We give a simple but robust algorithm for date extraction using the JodaTime [14] Library. We adapted a state-of-the-art system for timex extraction in the news domain (namely, HeidelTime) for using it in clinical narratives. We also developed our own rules for timex extraction to complement the HeidelTime and show that our rules give substantial performance improvement for this task.

2. Event extraction task

Task Description: In this task, we need to extract medical events from the clinical text. The i2b2 2012 challenge guidelines [1] defined a medical event as anything that is clinically important and that can also be mapped to a timeline. In addition to finding the text-spans referring to the events, we also need to identify these 3 attributes of the events: Type, Modality and Polarity. There are 6 possible values for the Type attribute: (1) laboratory tests, etc. (TEST), (2) treatments, medications, surgeries, etc. (TRE), (3) symptoms, diseases, complaints, etc. (PROB), (4) clinical departments like ICU (CDEPT), (5) evidential events like reporting a pain (EVID) and (6) any event which does not belong to first 5 types is referred to as Occurrence event (OCCU). The polarity attribute marks whether an event is positive (POS) or negative (NEG). For example, in the sentence "the patient reports headache, and denies chills", the event [headache] is positive in its polarity, and the event [chills] is negative in its polarity. The modality attribute is used to describe whether an event actually occurred or not. It can take four possible values: Factual (FACT), Conditional (COND), Possible (POS) and Proposed (PROP).

Approach Used: We used a pipeline approach for event extraction as illustrated in Fig. 1. In this approach, we first identify the spans of the events. These spans are given as input to the type classifier which finds the probability associated with any event taking a particular type. Then the event spans and type probability vectors serve as input for the modality and polarity classifiers which output the probabilities associated with any event taking particular modality and polarity values. The probability outputs of different classifiers are then given as input to the inference procedure which computes the final assignment of values to different event attributes. The different stages of this pipeline are described in more detail in the following subsections:

2.1. Finding the spans

We used a simple high-recall strategy for finding the spans of the events. Here, we take all the constituents[1] output by a state-of-the-art shallow parser[2] as candidate events. This list of candidates is then filtered in the following way:

1. The constituents which do not contain any letter (a–z) (e.g. 09/01/2011) are discarded.
2. Constituents which correspond to names of persons (e.g. Dr. Barber) are discarded.
3. Constituents corresponding to pronouns (e.g. he, she, etc.) are discarded.
4. Constituents which only consist of stopwords are discarded.

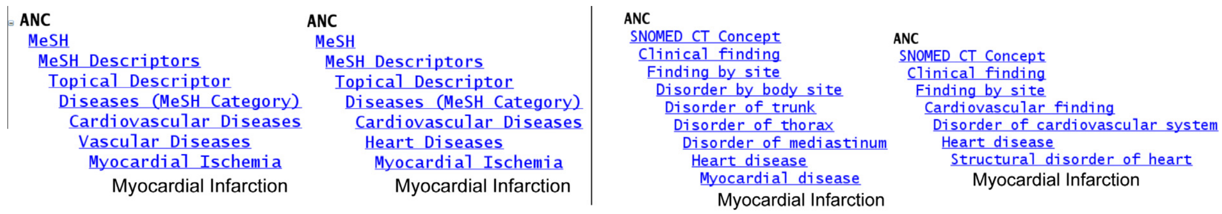
It is to be noted that a constituent which passes all the tests of this stage can still be rejected later on by assigning the type "NULL" to it. This justifies the use of high-recall strategy in this stage.

2.2. Finding the types of events

To find the types of events, we employed a multi-class SVM classifier as provided in the LIBSVM package [15]. Many of the constituents returned by the filtering process described above are still not events. So, we added a NULL type to the classifier to identify the constituents which are not events. Following features were used in the classifier design:

1. *Normalized string*: We normalized the surface form of the constituent by converting it to lowercase and removing the stopwords from it.
2. *Unigrams*: We split the surface form of the constituent to get the white-space delimited tokens. These tokens (other than those which correspond to stopwords) were used as the features.
3. *Semantic Type*: We processed the constituent using MetaMap which gives the UMLS concepts corresponding to the constituent. We used 2012AA USA-strict model in the MetaMap. Semantic types of these concepts were used as features.
4. *Occurrence in MeSH*: This is a binary feature which is active only if at least 1 of the concepts returned by MetaMap also occurs in the MeSH[3] Vocabulary.
5. *Occurrence in SNOMED CT*: This is a binary feature which is active only if at least 1 of the concepts returned by MetaMap also occurs in SNOMED CT[4] vocabulary.
6. *MeSH Descriptor*: MeSH Descriptor for a concept is found in the following way:
   (a) First of all, we construct the path from the concept to the root of the MeSH hierarchy. In general, there can be more than 1 paths. Fig. 2a and b show the different paths in MeSH

[1] Constituents refer to the chunks output by shallow parser.
[2] http://cogcomp.cs.illinois.edu/page/software_view/Chunker.
[3] http://www.nlm.nih.gov/mesh/meshhome.html.
[4] http://www.ihtsdo.org/snomed-ct/.



(a) Two different paths in MeSH Parent tree (b) Two different paths in SNOMED CT Parent tree

Fig. 2. Different paths in MeSH and SNOMED CT parent trees for the same concept "Myocardial Infarction".

and SNOMED CT parent trees for the same concept "Myocardial Infarction".

(b) For each path found in Step (a), we retain the top 4 concepts[5] (starting from the root of MeSH hierarchy). For example, for the two paths shown in Fig. 2a, we get the following 2 lists:
- (Diseases, Cardiovascular Diseases, Vascular Diseases, Myocardial Ischemia) for first path and
- (Diseases, Cardiovascular Diseases, Heart Diseases, Myocardial Ischemia) for second path.

It is to be noted that these lists differ in only the third member. Please also note that while making these lists, we did not include the following concepts: (MeSH, MeSH Descriptors, Topical Descriptor). This is because of the fact that these concepts appear at the top of almost all the paths and thus, do not provide any specific information.

(c) The lists obtained in previous step are then merged into a single list. For example, by merging the 2 lists mentioned in Step (b) above, we will get a single list with 8 members (some of which are duplicates). The duplicates are then removed but we keep track of the frequency with which each concept appeared in the list.

(d) The list obtained in step (c) is then sorted by concept frequency.

(e) Finally, we retain only the first 3 members of the list mentioned in Step (d). If there is a tie, preference is given to those concepts which are lower in the hierarchy because they often contain more useful information.

(f) The final list thus obtained in previous step gives us the MeSH descriptor of the concept.

MeSH descriptor of the full constituent is formed by taking the union of the MeSH descriptors of all the concepts occurring in the constituent as returned by MetaMap.

7. *SNOMED CT Descriptor*: SNOMED CT descriptor of the constituent is formed by a similar method as described for the MeSH descriptor.

Most of the features described above are lexical features which leads to a large number of total number of features. We get a total of 25,284 features and 31,291 training examples for our type classifier. SVM is able to effectively learn from such large number of features.

2.3. Finding the polarity of events

To find the polarity of events, we designed a binary classifier with the following features:

1. *Negex*: We use output of a publicly available implementation of Negex [16] algorithm as one of our features. Negex is a rule-based classifier for negation detection and the rules are specially designed for medical domain.
2. Unigrams from the constituent.
3. Four tokens from both left and right sides of the constituent.
4. Event type with the maximum probability as output by the type classifier previously.

2.4. Finding the modality of events

To find the modality of the events, we used a multi-class classifier using LIBSVM package. It used the same features as those in the polarity classifier except the NegEx feature.

2.5. Inference strategy for event extraction

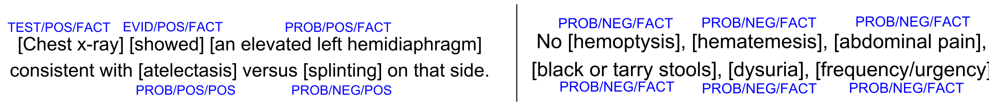
In this section, we describe our inference strategy for event extraction. The basic principle behind this strategy is: "*Attributes of events which appear close to one another are sometimes closely related. For some events, attributes can be determined with more confidence. And relations between events' attributes guide the inference procedure to determine the attributes of other events.*" We will now explain it in more detail with the help of examples. Fig. 3 shows two sentences in which the events are shown in brackets and correct (gold) attributes of events are shown adjacent to them. Event attributes are shown in the following order: Event Type/Event Polarity/Event Modality.

Now, consider first and third events in Fig. 3a. These events follow the pattern: [Event1] showed [Event2]. Let us call this pattern P1. In clinical narratives, such a pattern strongly suggests that Event1 is some kind of test and Event2 is some kind of problem (or symptom). So, we can impose a constraint on the output of classifiers that whenever it sees that two events follow pattern P1, then Event1 should be assigned the type TEST and Event2 should be assigned the type PROB.

Next, consider different events in Fig. 3b. All these events are separated by commas and hence, form a list. It is highly likely that all the events which appear in a list should have the same type. The advantage of such a constraint can be explained as follows: Suppose that type classifier is not sure about the type of second event "hematemesis" (which means that it does not give a high probability to any of the event types). But the type classifier is confident that first and third events ("hemoptysis" and "abdominal pain") are both problems. Based on the constraint that all the elements in a list should have the same type, type classifier can correctly infer that "hematemesis" should also be of type PROB.

It is also to be noted that each constraint can either be coded as a hard-constraint or a soft-constraint. If a constraint is generally not violated in the training data, then it is formulated as a hard-constraint. Otherwise, the constraint is made soft with a penalty parameter which is inversely proportional to the probability of violation of constraint in the training data.

[5] Let 'x' be the number of concepts to retain. This number was determined empirically. We were guided by the consideration that the retained concepts should neither be too general nor too specific. We experimented with different values of x. By manual observation, $x = 4$ gave the best results.



(a) Example 1 (b) Example 2

Fig. 3. Different events which appear in the same sentence constrain each other. In part (a), the pattern (Event1 showed Event2) indicates that Event1 is of type TEST and Event2 is of type PROB. Part (b) of the figure shows that events which appear in a list tend to have similar attributes (type, modality and polarity).

2.5.1. Implementing inference using Integer Quadratic Program (IQP)

In this section, we explain how to implement the inference procedure outlined in the previous section. We implement the inference procedure by writing it in terms of an optimization problem (specifically, Integer Quadratic Program (IQP)) as shown below in Eqs. (1)–(7). In this optimization problem, we basically try to assign those values to the events' attributes which get maximum probability according to the classifiers subject to several hard and soft constraints. To solve the IQP, we used Gurobi Optimizer 5.0 [17]. Previously, Roth and Yih [18] also suggested a formalism that combines constraints with linear models. However, unlike us, they restricted themselves to hard constraints.

$$\max \sum_{i=1}^{d}\left(\sum_{k=1}^{7} x_{type,i,k}\cdot p_{type,i,k} + \sum_{k=1}^{4} x_{mod,i,k}\cdot p_{mod,i,k} + \sum_{k=1}^{2} x_{pol,i,k}\cdot p_{pol,i,k}\right) - \sum_{(i,j)\in L}\rho_{type}\cdot \mathbb{1}\{z_{type,i}\neq z_{type,j}\} - \sum_{(i,j)\in L}\rho_{mod}\cdot \mathbb{1}\{z_{mod,i}\neq z_{mod,j}\} - \sum_{(i,j)\in L}\rho_{pol}\cdot \mathbb{1}\{z_{pol,i}\neq z_{pol,j}\} \tag{1}$$

$$\text{s.t. } \forall c\in C1: \forall i\in S_c: \bigvee_{k\in L_c} x_{type,i,k} \tag{2}$$

$$\forall c\in C2: \forall (i,j)\in S_c: \left(\bigvee_{k\in L1_c} x_{type,i,k}\right)\wedge\left(\bigvee_{k\in L2_c} x_{type,j,k}\right) \tag{3}$$

$$\forall c\in C3: \forall (i,j)\in S_c: \forall k: x_{type,i,k} == x_{type,j,k} \tag{4}$$

$$\sum_{k=1}^{7} x_{type,i,k} = 1, \sum_{k=1}^{4} x_{mod,i,k} = 1, \sum_{k=1}^{2} x_{pol,i,k} = 1 \quad \forall i \tag{5}$$

$$z_{type,i} = k \iff x_{type,i,k} = 1, z_{mod,i} = k \iff x_{mod,i,k} = 1, z_{pol,i} = k \iff x_{pol,i,k} = 1 \forall i \tag{6}$$

$$x_{type,i,k}\in\{0,1\}, x_{mod,i,k}\in\{0,1\}, x_{pol,i,k}\in\{0,1\} \quad \forall i,k \tag{7}$$

Eq. (1) describes the objective function of IQP. In this equation, $x_{type,i,k}$ denotes the indicator variable (as shown in Eq. (7)) which is equal to 1 (active) iff event i has type k. $z_{type,i}$ is another variable which takes an integer value (from 1 to 7) corresponding to the type of event i as shown in Eq. (6). $x_{mod,i,k}$, $x_{pol,i,k}$, $z_{mod,i}$ and $z_{pol,i}$ are similarly defined. $p_{type,i,k}$ is the probability of event i having the type k as output by the type classifier. $p_{mod,i,k}$ and $p_{pol,i,k}$ are similarly defined.

Next, we describe the different constraints used in the optimization problem:

1. The last 3 summation terms in Eq. (1) correspond to three soft constraints with penalty parameters given by ρ_{type}, ρ_{mod} and ρ_{pol}. Here, L is the set of 2-tuples where each tuple contains ids of 2 events which appear adjacent to one another in a list. Thus, these constraints penalize those assignments where the events which appear in a list get different attributes.
2. $C1$ in Eq. (2) refers to the set of those constraints which act only on a single event. Here, S_c is the set of events which satisfy constraint c and L_c is the set which contains the possible event types for any event in S_c.
3. $C2$ in Eq. (3) is a set of binary constraints (constraints which take 2 events as arguments). S_c is the set of event tuples which satisfy constraint c. $L1_c$ and $L2_c$ are the sets of possible event types for the first and second element of event tuples in S_c. An example of this constraint is a pattern "[Event1] showed [Event2]" as shown in Fig. 3a.
4. $C3$ in Eq. (4) is another set of binary constraints which enforce the events in every tuple of S_c to have the same type. For example, any two events which are separated by a comma and are part of the same list should have same type.
5. Eq. (5) imposes the constraint that every event has a unique type, modality and polarity.

It is to be noted that the inference procedure as described above is very general and allows for several other types of constraints. For example, there can be constraints which relate the values of different attributes of the same (or even different) event(s). However, we have not yet experimented with such constraints and we will address this issue in future work.

3. Timex extraction

Task Description: In the Timex extraction task, a system is supposed to identify the spans and attributes of the temporal expressions in the text. The i2b2 2012 challenge guidelines [1] defined three attributes associated with each temporal expression, namely type (TYPE), value (VAL) and modifier (MOD). The TYPE attribute can have 4 possible values: DATE, TIME, Duration (DUR) and Frequency (FREQ). The VAL attribute gives the time (value) associated with the temporal expression. Finally, a temporal expression can have one of the following 7 modes: NA, APPROX, MORE, LESS, START, MIDDLE, END.

Approach Used: Our overall approach for timex extraction is rule-based as rule-based methods have been shown to give the best results to date for this task [11]. For the timex extraction task, first of all we determine the "Admission Date" and "Discharge Date" as given in the clinical narrative. Then we use a publicly available temporal extraction system, namely HeidelTime, as our basic building block. And finally, we use our own rules which are specifically designed for clinical narratives to complement the output of HeidelTime. We explain each of these components of our temporal extraction system in the following subsections.

3.1. Finding section times

In this subsection, we describe the method used by us to determine "Admission Date" and "Discharge Date" in the clinical narrative.

1. Clinical narratives in the i2b2 datasets typically have 4 sections: Admission Date, Discharge Date, History of Present Illness and Hospital Course. A new section is determined by the fact that the line ends with a semicolon.
2. After we determine the first line where "Admission Date" and "Discharge Date" sections begin, we use a regular expression to find out whether the following line has a date in it.
3. Now, a date can be written in several different formats. For example, we can write the same date Sep 14, 1999 in the

following ways: 1999-09-14, 99/09/14, 09/14/99, 09/14/1999, 09-14, etc. Clinical reports list the dates in all such formats. Correctly determining the date requires consideration of the constraints on the date fields, namely day, month and year. For example,

(a) Since the narratives were all taken from US hospitals, they put month before date.
(b) Month takes value between 1-12 and day takes value between 1-31.
(c) Year can appear either as a first field or a last field.
(d) Year can either be in 2 digit format or in 4 digit format.
(e) It is possible that year may not be there at all (as it can be determined from context). But since we are dealing with clinical reports, the day field will generally be there.
(f) Both "-" and "/" can act as separators between various fields of date expression.

4. Considering the above things, we designed the following 10 regular expressions using JodaTime Library: yy-MM-dd, MM-dd-yy, yyyy-MM-dd, MM-dd-yyyy, yy/MM/dd, MM/dd/yy, yyyy/MM/dd, MM/dd/yyyy, MM-dd, MM/dd. The given date expression was made to match with each of these regular expressions. JodaTime itself takes care of consistency checks on the date fields. First regular expression which matched the date expression was used to determine the date fields. This algorithm gave us an accuracy[6] of 97% on test portion of the i2b2 2012 dataset.

The above procedure was also used to determine dates in other sections of the clinical narrative.

3.2. Using HeidelTime

1. We used HeidelTime as a basic building block to obtain the temporal expressions. For the temporal expressions that HeidelTime identifies, it also gives the TYPE, VAL and MOD attributes. HeidelTime also expects the "Document (Section) Creation Time (DCT)" as one of the inputs which serves to resolve the ambiguity while determining the value of some temporal expressions. For "History of Present Illness" section, DCT is given to be the Admission date and for "Hospital course" section, DCT is given to be the Discharge Date.
2. HeidelTime assigns the type SET to timexes of type FREQ. So, we replace the SET type in HeidelTime with FREQ while outputting the result. Also, in the VAL attribute for timexes of type FREQ, HeidelTime does not prefix the value with R. So, we ourselves prefix the value of FREQ timexes with "R". In some cases, the VAL attribute given by HeidelTime is not formatted according to the i2b2 guidelines. So, we do a post-processing step to properly format the HeidelTime results. For example, for the phrase "several months", HeidelTime gives the value *PXM*.[7] We change it to *P3M* and set the modifier attribute to *APPROX* as specified in the i2b2 guidelines.
3. Next, the MOD attribute produced by HeidelTime is mapped to the i2b2 MOD attribute. For example, "more_than" of HeidelTime is same as "more" of i2b2.

3.3. Rules for identifying clinical timexes

Although HeidelTime is good in identifying timexes written in general English, it is not able to identify the clinical timexes. So, we added the following rules in our system to identify the clinical timexes.

1. Some timexes are of the form "POD#*n*".[8] For such timexes, first we identify the temporal expression corresponding to the "operation date". If we do not find such an expression, then we set the operation date to be same as admission date. Next, we set the value of "POD#*n*" to *n* days after the operation date. We also capture other variations of "POD#*n*" like "postoperative day *n*", etc.
2. A similar procedure as described above is also followed for the expressions like "hospital day *n*" or "HD *n*". Reference date for computing the value of such timexes is the admission date.
3. We identify clinical expressions of the type "x *n*" or "x*n*" or "times *n*", etc. Such expressions are of type frequency and identify only the number of times for which an event is repeated but do not specify the interval for such repetition. A value of "R*n*" is given to these expressions.
4. If year value is not specified for some of the dates mentioned within the document, then we set the year based on admission date or discharge date.
5. Several timexes contain the word "day" in them. Expressions like "per day" are assigned the type FREQ with a period of 1 day. For expressions like "2 days after admission (discharge)", we calculate the value based on admission (discharge) date and assign the type DATE to such expressions.
6. Expressions like "tid" or "t.i.d.", etc. are of type FREQ with period of 8 h. Similar rules are also developed for expressions of type "bid". These expressions have the period of 12 h.
7. Several timexes start with the letter "q". We developed rules for all such expressions. For example, our rules cover the following expressions: qid (Period: 6 h), qad (Period: 48 h), qd (Period: 24 h), qds (Period: 6 h), qAM or qPM (Period: 24 h), qn or qnoc ("every night", Period: 24 h), qmt ("every month", Period: 1 month), qw ("every week", Period: 1 week), etc. Please note that our rules also cover variations of such expressions as well. Expressions of type "q*n*h" have the period of *n* hours where *n* is a number.

4. Experiments and results

4.1. Datasets

For our experiments, we used the data provided by the i2b2 team as part of i2b2 2012 shared task [1]. The input consists of plain text files and the output consists of event and timex annotations along with their respective attributes. Training data has a total of 190 records and the test data has 120 records.

4.2. Event extraction

In Table 1a, we report precision (P), recall (R) and F1 scores of our system on the test set. P, R and F1 scores are reported under 3 separate headings: Extent-Only, Extent + Type and Extent + All Attributes. For extent-only heading, a predicted event is considered to be correct if its extent overlaps with the extent of some gold event (i.e. we use lenient matching). For Extent + Type heading, the type attribute of predicted event should also match the type attribute of gold event. Similarly, for Extent + All Attr heading, all attributes (type, modality and polarity) of predicted event should match that of gold event.

First 6 rows of Table 1a show the P, R and F1 scores for the individual event types and the last row reports these scores for all the categories combined. Overall, we obtained 0.87, 0.77 and 0.71 F1 scores for Extent-Only, Extent + Type and Extent + All Attr matching criteria respectively. Thus, there is a drop of 0.10 in F1 score because of the errors made in finding the type of the events and there

[6] Here, accuracy refers to the fraction with which our system correctly identifies the date associated with a date pattern. For example, the pattern "7/12/99" represents the date "July 12, 1999".

[7] Here, "X" represents an unknown value.

[8] "POD" stands for "postoperative day".

Table 1
Two tables in part (a) and part (b) show the results for event extraction and timex extraction tasks respectively. P and R in these tables refer to Precision and Recall respectively. In part (b), ST stands for Section Times and HT stands for HeidelTime. Last 3 rows in part (b) report the accuracy values for TYPE, VAL and MOD attributes.

	Extent-Only			Extent + Type			Extent + All Attr		
	P	R	F1	P	R	F1	P	R	F1
(a) *Event extraction task*									
TEST	0.96	0.87	0.91	0.85	0.79	0.82	0.79	0.74	0.77
TRE	0.93	0.86	0.89	0.85	0.78	0.81	0.78	0.71	0.74
PROB	0.91	0.88	0.90	0.81	0.84	0.82	0.71	0.73	0.72
CDEPT	0.87	0.81	0.84	0.81	0.76	0.78	0.80	0.74	0.77
EVID	0.87	0.74	0.80	0.81	0.60	0.69	0.80	0.60	0.69
OCCU	0.81	0.75	0.78	0.68	0.66	0.67	0.65	0.64	0.65
Overall	0.90	0.84	0.87	0.80	0.75	0.77	0.73	0.69	0.71

	ST	+HT	+Rules
(b) *Timex extraction task*			
P	1.00	0.84	0.83
R	0.13	0.54	0.76
F1	0.22	0.66	0.79
TYPE	0.13	0.50	0.71
VAL	0.12	0.41	0.56
MOD	0.13	0.49	0.70

is a further drop of 0.06 because of errors made in finding the other attributes (modality and polarity). The overall drop in F1 score for the individual categories is 0.14, 0.15, 0.18, 0.07, 0.11 and 0.13 for TEST, TRE, PROB, CDEPT, EVID and OCCU respectively. Thus, PROB category experienced the maximum drop in F1 score because of the errors made in finding the attributes.

For Extent-only and Extent + Type matching criteria, TEST, TRE and PROB categories gave the best results (subject to small difference of 0.01 or 0.02) whereas for Extent + All Attributes matching criterion, CDEPT and TEST categories gave the best results.

In addition, please also note that the accuracies of our system for type, modality and polarity attributes for correctly predicted events are 0.74, 0.77 and 0.75 respectively as given by the i2b2 2012 challenge official evaluation script.

4.2.1. Impact of IQP inference

IQP inference procedure used by us gave marginal improvement for event extraction task. Different categories benefitted to different extents from the inference procedure as explained below:

- CDEPT, EVID and OCCU: IQP inference had no effect on event extraction performance for these three categories. This is because of the fact that the constraints used by us did not cover these three categories.
- TEST, TRE and PROB: For TEST, TRE and PROB categories, we obtained small improvements of 0.1%, 0.2% and 0.3% respectively. For example, for PROB category, the performance improved from 0.719 to 0.722. According to Koehn's bootstrap resampling test [19], the performance improvement is statistically significant (at $p = 0.05$) only for TRE and PROB categories.

4.3. Timex extraction

Table 1b gives the Precision (P), Recall (R) and F1 scores for the timex extraction task. For evaluating these scores, a predicted temporal expression is considered to be correct if its extent overlaps with that of some gold temporal expression. Last 3 rows in Table 1b report the accuracy values for TYPE, VAL and MOD attributes. Accuracy for attributes is determined by the fraction of correctly predicted timexes whose attributes match the attributes of gold timexes.

In the first column in Table 1b, we report scores for the case where we only find the section times (ST) i.e. admission and discharge date. Second column reports scores for the case where we also use HeidelTime (HT) in addition to finding section times. And the last column reports the scores for the case when the full system is used. We see that F1 score increases by 0.44 as a result of using HeidelTime. Addition of rules developed by us leads to a further increase of 0.13 in F1 score. Similar improvements can also be seen for TYPE, VAL and MOD attributes.

4.4. Comparison with other i2b2 2012 systems

The i2b2 2012 challenge attracted a lot of participation [20–26] from all over the world. Sun et al. [1] present an overview of the 2012 i2b2 shared task. For event extraction, our system was ranked 10th among all the systems that participated in the shared task. We lagged behind the top performing system [22] by 0.09 F1 points. For detailed comparison of the systems, please refer to Sun et al.

For timex extraction, the primary evaluation metric was the product of F1 score and the accuracy of VAL attribute. For our system, this score is 0.44 which is just 0.01 points behind the system which was ranked 10th in the shared task [25]. For timex extraction, we lagged behind the top performing system [20] by 0.22 F1 points.

5. Error analysis

5.1. Event extraction

Table 2 shows the error analysis for event extraction task. The columns in Table 2 correspond to correct (gold) event types and rows correspond to predicted event types. If the entry under column *b* and row *a* is *n*, it means that *n* events whose correct type was *b* were wrongly predicted to have the type *a*.

In Table 2, the last column (except the entry for NULL) corresponds to precision errors and the last row (except the entry for NULL) corresponds to recall errors. We see that we make more recall errors than precision errors.

Error analysis of event extraction system revealed the following major sources of errors:

1. Among all the event types, maximum number of errors are made for the type "Occurrence (OCCU)". These account for roughly 19% false negatives. This was expected because definition of OCCU is quite vague: Any event which does not belong to first 5 types is assigned the type OCCU.
2. Our system identifies many clinical events like "bowel movements", "flatus", etc. which are not annotated in the gold standard. These account for about 21% false positives. So, one possible way to improve the accuracy of our system is to add an extra stage in our pipeline shown in Fig. 1 which exclusively separates NULL events from rest of the events.
3. Incorrect span determination is another major source of false positives (about 24%). Incorrect span determination further leads to errors in predicting the type of the events. For example,

Table 2
Error analysis for the event types.

	TEST	TRE	PROB	CDEPT	EVID	OCCU	NULL	TOTAL
TEST	–	44	99	4	1	39	77	264
TRE	49	–	58	7	2	45	190	351
PROB	96	91	–	9	2	120	370	688
CDEPT	2	7	2	–	0	23	90	124
EVID	1	0	3	1	–	10	57	72
OCCU	23	89	88	13	61	–	465	739
NULL	285	461	482	140	148	604	–	2120
TOTAL	456	692	732	174	214	841	1249	–

in one case, our system identified the span as "GI bleed" whereas the gold annotation was "GI bleed precautions".

4. For some events, predicting the attributes correctly requires significant understanding of the surrounding context. For example, "blood cultures" event is generally of type TEST but if blood cultures have been determined to be positive, then this event should be classified as a PROB. Similarly, "bicarbonate" event can either be a test or a treatment depending on the context. Our system currently does not perform deep semantic analysis.
5. Lack of world knowledge (like unseen events, unknown abbreviations (NKA, TPN, etc.)) is a major source of false negatives (about 23%). Availability of more training data will help to address this problem.

5.2. Timex extraction

A manual analysis of the errors made by timex extraction system revealed the following major sources of errors:

1. *Ambiguous expressions*: Some of the temporal expressions (like "this time", "that time") were quite ambiguous. These expressions sometimes need to be annotated and sometimes not. Our system made some errors (about 14% false positives) on such expressions.
2. *Lesser known cases*: Some of the expressions like "night PTA", "three cycles", "a couple weeks", "the ensuing days", etc. were not so common in the training data. So, our rules did not cover such expressions. These accounted for about 26% false negatives.
3. *Complex expressions*: Some of the expressions were quite complex and difficult to parse and interpret. Examples of such expressions are: "day of life#1", "Q4-6H", "time of transfer", "30-6/7 weeks", "day of delivery", etc. Our system failed to identify such expressions resulting in about 13% false negatives. These errors can be handled by formulating more rules.
4. *Partially correct*: For some of the events, the value determined by us was only partially correct. For example, for the expression "12:30 on 2000-04-02", our system obtained the value "T12:30" but the correct value also included the date in it. Similarly, for the expression "bid for 1 day", our system obtained the value "RPT12H" but the correct value was "R2PT12H".
5. Our system wrongly identified some of the date-like expressions (e.g. 11/21/70) as valid temporal expressions but such expressions were actually test results. These accounted for about 19% false positives. These errors can be handled by semantic interpretation of the surrounding context.

6. Conclusion

This paper addressed the task of event and timex extraction from clinical narratives. For the event extraction task, we developed a sentence-level inference strategy which exploits relationship between related events. Our inference strategy is general and it can also be used for performing inference among different tasks. Clinical descriptors designed by us contain useful information about concepts and have broad applicability. In future work, we would explore the use of machine-learning methods for the timex extraction task which would reduce the need for manually specifying the rules or which would simplify the process of making the rules.

Acknowledgments

The authors thank the anonymous reviewers for their valuable suggestions. This research was supported by Grant HHS 90TR0003/01 and by the Intelligence Advanced Research Projects Activity (IARPA) Foresight and Understanding from Scientific Exposition (FUSE) Program via Department of Interior National Business Center contract number D11PC2015. In addition, the i2b2 challenge was supported by Grants NIH NLM 2U54LM008748 (PI: Isaac Kohane) and NIH NLM 1R13LM011411-01 (PI: Ozlem Uzuner). The contents of this paper are solely the responsibility of the authors and do not necessarily represent the official views of the HHS, IARPA, DoI/NBC, NIH, NLM or the US government.

References

[1] Sun W, Rumshisky A, Uzuner O. Evaluating temporal relations in clinical text: 2012 i2b2 challenge. J Am Med Inform Assoc 2013.
[2] de Bruijn B, Cherry C, Kiritchenko S, Martin J, Zhu X. Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010. J Am Med Inform Assoc 2011;18(5):557–62.
[3] Jiang M, Chen Y, Liu M, Rosenbloom S, Mani S, Denny J, et al. A study of machine-learning-based approaches to extract clinical entities and their assertions from discharge summaries. J Am Med Inform Assoc 2011;18(5):601–6.
[4] Roberts K, Harabagiu S. A flexible framework for deriving assertions from electronic medical records. J Am Med Inform Assoc 2011;18(5):568–73.
[5] Uzuner Ö, Zhang X, Sibanda T. Machine learning and rule-based approaches to assertion classification. J Am Med Inform Assoc 2009;16(1):109–15.
[6] D'Avolio L, Nguyen T, Goryachev S, Fiore L. Automated concept-level information extraction to reduce the need for custom software and rules development. J Am Med Inform Assoc 2011;18(5):607–13.
[7] Torii M, Wagholikar K, Liu H. Using machine learning for concept extraction on clinical documents from multiple data sources. J Am Med Inform Assoc 2011;18(5):580–7.
[8] Uzuner O, South B, Shen S, DuVall S. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. J Am Med Inform Assoc 2011.
[9] Strötgen J, Gertz M. Multilingual and cross-domain temporal tagging. Lang Resour Eval 2012:1–30.
[10] Chang AX, Manning CD. SUTIME: a library for recognizing and normalizing time expressions. In: 8th International conference on language resources and evaluation (LREC 2012); 2012. <http://nlp.stanford.edu/pubs/lrec2012-sutime.pdf>.
[11] Verhagen MSauri R, Caselli T, Pustejovsky J. Semeval-2010 task 13: Tempeval-2. In: Proceedings of the 5th international workshop on semantic evaluation. Association for Computational Linguistics; 2010. p. 57–62.
[12] Angeli G, Manning CD, Jurafsky D. Parsing time: learning to interpret time expressions. In: NAACL-HLT; 2012. <http://nlp.stanford.edu/pubs/2012-naacl-temporal.pdf>.
[13] Sun W, Rumshisky A, Uzuner O. Temporal reasoning over clinical text: the state of the art. J Am Med Inform Assoc 2013.
[14] JodaTime; 2013. <http://joda-time.sourceforge.net/> [accessed 10.06.13].
[15] Chang CC, Lin CJ. LIBSVM: a library for support vector machines. ACM Trans Intel Syst Technol 2011;2:27:1–27:27. <http://www.csie.ntu.edu.tw/cjlin/libsvm>.
[16] Chapman W, Bridewell W, Hanbury P, Cooper G, Buchanan B. A simple algorithm for identifying negated findings and diseases in discharge summaries. J Biomed Inform 2001;34(5):301–10.
[17] Gurobi Optimization I. Gurobi optimizer reference manual; 2013. <http://www.gurobi.com>.
[18] Roth D, Yih W. Global inference for entity and relation identification via a linear programming formulation; 2007. <http://cogcomp.cs.illinois.edu/papers/RothYi07.pdf>.
[19] Koehn P. Statistical significance tests for machine translation evaluation. In: Proceedings of EMNLP, vol. 4; 2004. p. 388–95.
[20] Sohn S, Wagholikar KB, Li D, Jonnalagadda SR, Tao C, Elayavilli RK, et al. Comprehensive temporal information detection from clinical text: medical events, time, and TLINK identification. J Am Med Inform Assoc 2013.
[21] Tang B, Wu Y, Jiang M, Chen Y, Denny JC, Xu H. A hybrid system for temporal information extraction from clinical text. J Am Med Inform Assoc 2013.
[22] Xu Y, Wang Y, Liu T, Tsujii J, Eric I, Chang C. An end-to-end system to identify temporal relation in discharge summaries: 2012 i2b2 challenge. J Am Med Inform Assoc 2013.
[23] Cherry C, Zhu X, Martin J, de Bruijn B. À la recherche du temps perdu: extracting temporal relations from medical text in the 2012 i2b2 NLP challenge. J Am Med Inform Assoc 2013.
[24] Roberts K, Rink B, Harabagiu SM. A flexible framework for recognizing events, temporal expressions, and temporal relations in clinical text. J Am Med Inform Assoc 2013.
[25] Grouin C, Grabar N, Hamon T, Rosset S, Tannier X, Zweigenbaum P. Eventual situations for timeline extraction from clinical reports. J Am Med Inform Assoc 2013.
[26] Kovačević A, Dehghan A, Filannino M, Keane JA, Nenadic G. Combining rules and machine learning for extraction of temporal expressions and events from clinical narratives. J Am Med Inform Assoc 2013.